In *V&V State of the Art: Proceedings of Foundations '02, a Workshop on Model and Simulation Verificatinon and Validation for the 21st Century*, Dale K. Pace (ed), held October 22-24, 2002, Johns Hopkins University Applied Physics Laboratory, Laurel, MD. Published by The Society for Modeling and Simulation International, www.scs.org, ISBN 1-56555-256-3 2002.

Infusing and Selecting V&V Activities

Martin S. Feather
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr
Pasadena CA 91109-8099
Martin.S.Feather@Jpl.Nasa.Gov

Abstract

The evolving nature of software development poses a continuing series of challenges for V&V. In response, the V&V community selectively adapts the use of existing V&V activities, and introduces new and improved ones.

These responses are instances of the more general issues of technology selection and technology infusion. These are recurring challenges at JPL, where novel spacecraft applications demand novel adaptations of existing technologies, and infusion of new ones. JPL has been developing and applying a process specifically to assist in the planning for these. This paper shows how this same process has the capacity to aid in planning the selection and infusion of V&V activities.

1. Introduction

Planning the quality assurance activities for spacecraft systems is very challenging. There are typically far more assurance activities (e.g., analyses, tests, inspections, reviews, standards, policies, certifications, defensive measures) from which to choose than there are resources (e.g., time, budget, testbeds, personnel) available to perform those assurance activities. Best practices call for the judicious selection of assurance activities, to make optimal use of the limited resources, is essential.

JPL deploys spacecraft in new and challenging situations, employing new technologies and designs to better attain mission objectives. This means that innovation is a recurring phenomenon – each spacecraft exhibits some new design aspects. As a result, the challenge for quality assurance planning is to adapt and extend best practices and lessons learned from past missions to each new spacecraft.

In response to this challenge, Dr. Steve Cornford at JPL conceived of a quantitative model specifically to facilitate assurance planning [Cornford 1998]. His model, called "Defect Detection and Prevention" (DDP), is designed for application early in the lifecycle, when information is sparse, yet the capability to influence the course of the development to follow is large. His initial experiments used Microsoft Excel® spreadsheets to manually explore the utility of the process. Positive results from these led to an effort to develop custom software for the DDP process [Feather et al, 2000a]. Supported by this software, DDP has been applied to assess the viability of, and planning for, the development of novel technologies and systems [Cornford et al, 2001], [Cornford et al, 2002].

The core idea of DDP is to relate three sets of information: the objectives you want to achieve, the problems that can get in the way of attaining those objectives, and the options you have to overcome those problems. In application to design and development planning for spacecraft, DDP has evolved to calling these three sets of information "*Objectives*" (alternately "Requirements"), "*Risks*" (for studies of hardware, these have also been referred to "Failure Modes") and "*Mitigations*" (to emphasize their effect of preventing, reducing and/or alleviating risks; in previous DDP papers referred to via the acronym PACTs - an acronym of **P**reventions, **A**nalyses, process **C**ontrols and **T**ests). A further important characteristic of DDP is its *quantitative* treatment of the relationships between information (e.g., *how much* a Risk, should it occur, detracts from an Objective's attainment). This quantitative treatment is key to DDP's realization of the vision of "risk as a resource", as espoused in [Greenfield, 1998].

The focus of this paper is to show the potential applicability of DDP to two important aspects of V&V:

1. Furthering the infusion of promising new V&V techniques.
2. Planning the judicious selection of V&V techniques for novel and challenging applications.

Note that both these areas involve a novel element. In contrast, when well-understood V&V techniques are to be applied to a well-understood problem area, it is likely that best practices will already have been established. In such circumstances (e.g., "product line" developments) the key challenge is one of ensuring the systematic and disciplined application of V&V, for which process-centric approaches such as ISO and CMM are eminently well suited.

The remainder of this paper is organized as follows:

Section 2 presents details of DDP's quantitative model.

Section 3 illustrates the use of this model to study the technology infusion of an advanced V&V technique.

Section 4 describes our approach to using DDP for V&V planning.

2. DDP's Quantitative Risk-based Model

As introduced in the previous section, the core idea of DDP is to relate "Objectives" (what you want to achieve), "Risks" (what problems can get in the way of attaining those Objectives), and “Mitigations” (what you can choose to do to overcome the problems). The subsections that follow present this model in more detail.

2.1. DDP Motivation

Motivation for the DDP model stems from Cornford’s original vision of development activities *filtering out* risk during spacecraft development (Fig 1). Risks are items

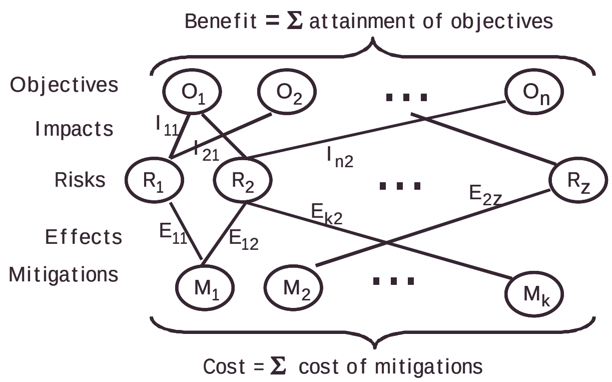

Figure 2. Topology of DDP model

whose presence threatens mission success. They are filtered out by the various development activities, shown as the intermediate rectangular boxes in the figure (note: these are not drawn to scale!). Risks that escape such filtering threaten mission success. For illustrative purposes, three Risks have been highlighted.

- The green-colored Risk passes through several boxes, indicating multiple opportunities to quell that particular Risk. However, these filtering activities consume scarce resources. Hence, it might be the case that this particular Risk is *over*-filtered, and the resources expended on its filtering might be better used for other purposes.
- The blue-colored Risk is filtered by one and only one activity. Depending upon the circumstances, this might or might not be sufficient. For example, if this is a particularly severe risk (with potential to cause loss of the entire mission, say), then it might be prudent to employ more than just one activity to filter it out.
- Lastly, the red-colored Risk is completely unfiltered. Chance alone will determine whether it impacts mission success.

DDP is designed to represent in a detailed fashion the information represented in this figure. Risks are the entities whose presence threatens mission success. Mitigations are the activities that filter out Risks to varying degrees. Risks are not all equal – those that threaten more important mission objectives to greater extents and with greater likelihood are the more severe ones. Likewise, Mitigations are not all equal – a given Mitigation may filter out one Risk more effectively than another.

The overall topology of a DDP model is sketched in Figure 2. This conveys the possibilities that multiple Risks can impact an Objective, a Risk can impact multiple Goals, a Mitigation can effect multiple Risks, and multiple Mitigations can effect a Risk. Bear in mind that in DDP these are not simply boolean relationships: the links indicating these impacts and effects have associated numerical values, indicating the *strength* of the relationship.

Overall, this is a relatively straightforward model, involving only three types of concepts, and two types of links. Some additional refinements of the model to capture further nuances will be described in section 4. In DDP applications to date most of the information that has been captured has fit into the simple model sketched above. However, the information is typically quite voluminous, as seen in Figure 3, showing a real DDP application’s information drawn in this same style.

In practice the DDP software uses alternate views of this information more palatable to the human viewers. For example, DDP makes use of hierarchy to organize information into meaningful taxonomies, which are much more amenable to scrutiny, search and comprehension. Some illustrations will appear in the sections that follow; for additional information, the reader is referred to [Feather et al, 2000a].

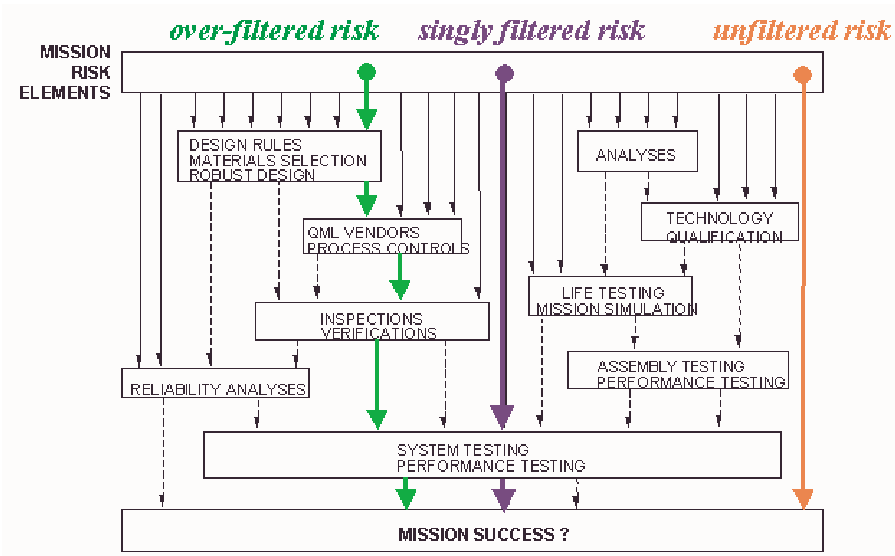

Figure 1. Development Activities *Filter Out* Risks

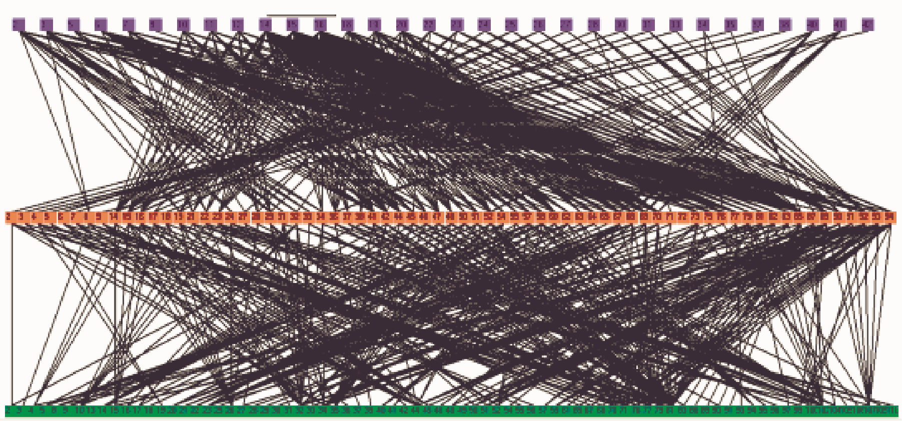

Figure 3. Topology of the data in an actual DDP model

2.2. DDP core concepts

- "**Objectives**" – the things that the system is to achieve, and the limitations within which it must operate. Objectives can be assigned different "weights" to reflect their relative importance.
- "**Risks**" – all the kinds of things that, should they occur, would lead to failure to attain Objectives.
- "**Mitigations**" – all the things that could be applied to reduce Risks. These could be preventative measures, tests, analyses, inspections, reviews, redundant design elements, etc. They achieve their beneficial effect by:
  - *preventing* Risks from occurring in the first place (decreasing their likelihood),
  - *detecting* the presence of Risks prior to use, thus allowing for the opportunity to repair/correct such problems (for spacecraft, hardware repair must usually be done prior to launch, while repairs to software and changes to operating procedures can be done after launch), or
  - *alleviating* the severity of Risks should they occur.

  Mitigations have associated costs, which may be in multiple dimensions (e.g., schedule, budget, personnel time, testbed resources, mass, power)

2.3. DDP relationships

Objectives, Risks and Mitigations are *quantitatively* related to one another in the following manner:

- Objectives are quantitatively related to Risks, to indicate the proportion of the Objective attainment that would be lost should the Risk occur. In DDP terminology, we say that a Risk has an "impact" on an Objective.
- Risks are quantitatively related to Mitigations, to indicate the proportion by which each Mitigation reduces each Risk should that Mitigation be applied. In DDP terminology, we say that a Mitigation has an "effect" on a Risk.

The model further assumes that:

- Risks' impacts on an Objective combine by addition. For example, if two Risk have impacts on the same Objective of 0.1 and 0.2, then their combined impact is 0.1 + 0.2 = 0.3.
- Mitigations' effects on Risks combine by, essentially, multiplication of their complements. For example, if two Mitigations have effects on the same Risk of 0.1 and 0.2, then their combined effect is (1 – (1 – 0.1)*(1 – 0.2)) = 0.28. The intuition behind this is that Mitigations *filter* Risks, and that multiple Mitigations act like filters in series.

On occasion, there is a mismatch between the case at hand and the combination rules assumed by the model. Often, such mismatches can be handled by manual workarounds. For example, if the combination of two Mitigations is not as effective as the model's combination rule would calculate, enter a third Mitigation to represent that combination, manually score its effects accordingly, and thereafter be careful to select at most one of those three Mitigations (either of the individual ones, or this manually scored combination one).

2.4. Quantitative Reasoning in DDP

One of the hallmarks of DDP is its focus on early-lifecycle application using a *quantitative* basis. Other approaches to reasoning at such early stages generally resort to *qualitative* treatments. Those techniques that do adopt a qualitative basis often require a detailed design on which to base their reasoning (e.g., fault tree analysis).

DDP aims to fill the niche between qualitative approaches and detailed design-centric analysis approaches. It relies heavily on expert estimates of qualitative effects (e.g., experts are asked to estimate the *proportion* of an Objective's attainment that will be lost if a Risk occurs). These figures need not be given to multiple digits of precision, but do need to go beyond merely ordered but non-quantitative rankings such as "high" "medium" and "low".

We have also investigated a more purely qualitative approach, which we embodied in our "Risk Balancing Profiles" (RBP) tool. This too involved the key idea of explicitly relating Risks to Mitigations, but did not distinguish between different strengths of such relationships. Instead, the tool simply indicated to users the Mitigations applicable to each Risk, and kept track of their choices. Populating this simple tool with preformulated taxonomies of Risks and Mitigations proved useful to remind people of this information. This was seen as a natural precursor to DDP, and we arranged to permit RBP's information to be transferable into DDP [Feather et al, 2000b]. In practice, users in situations of relative familiarity seem to prefer to make use of the standard checklist like features of RBP, while when faced with more novel challenges, turn to DDP to help them.

2.5. Quantitative calculations in DDP

DDP's quantitative nature makes possible automatic computation of several risk-related measures:

- The proportion to which an Objective is "at risk": computed by summing the impacts of Risks on that Objective. Note that this sum could exceed 1, which at first sight appears ridiculous – how can an Objective be more than totally at risk? A simple example would be a situation in which multiple fatal flaws exist, any one of which would alone lead to complete loss of an Objective. The "at risk" measure gives an indication of the amount of corrections that need to be done to attain that Objective. Compare that situation with one in which only one fatal flaw exists. In either situation, the Objective is not attained. However, in the latter case, there is but one flaw to correct, which would seem to be a much better situation to be in.
- An Objective's "attainment" proportion: computed as (1 – minimum(1, its "at-risk" measure))
  The minimum calculation is there because, as discussed above, an Objective's "at risk" measure may exceed 1. Multiplying an Objective's attainment proportion by its weight (the user-assigned value denoting its relative importance) gives a measure of the benefit attained of that Objective.
- Overall Objectives attainment: computed by the summing the benefit attainment of all the Objectives. This is a single measure of the total "benefit" of a DDP model, and so is useful for comparing alternative selections of Mitigations.
- The extent to which a Risk is contributing to total risk: computed by summing over all the Objectives the Risk's impact on the Objective multiplied by the Objective's weight. This can be calculated either with or without taking into account the beneficial effects of chosen Mitigations. One of the tenets of DDP is to strive towards a somewhat balanced treatment of risk mitigation. There is little point seeking to further reduce an already miniscule risk when there remain other risks with comparatively greater magnitude.
- Overall cost of selected Mitigations: in the simple DDP model, computed by simply summing the resource costs of the selected Mitigations. More elaborate aspects of cost calculation come into play when taking into account the cost of repair of detected flaws (e.g., the cost of fixing a bug detected during testing). However, the overall cost for a DDP model is still a function of the selected Mitigations, and is used along with the calculation of Objectives' attainment to choose from among alternative Mitigation selections.

2.6. DDP user interfaces

DDP is used to assist, not replace, users in their decision-making. Thus the DDP user interface is crucial. Users must be able to enter information into DDP, survey the information they have provided, and scrutinize the ramifications of that accumulated information.

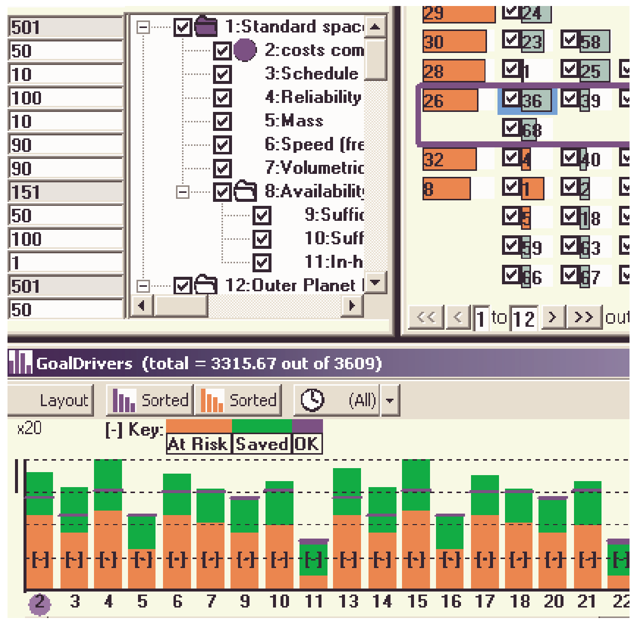

Figure 4. Fragment of a DDP screen

The sections that follow will illustrate some of the DDP user interface features. Briefly, the key visualizations are of:

- Trees to present the taxonomies of the core concepts (Objectives, Risks and Mitigations),
- Matrices to present the quantitative relationships between the core concepts (how much each Risk impacts each Objective; how much each Mitigation reduces each Risk).
- More compact forms of the (generally rather sparse) matrices.
- Bar charts to present an entire set of core concepts (e.g., a bar chart showing one bar for each Objective).
- A 2-d plot of all the Risks, where the dimensions indicate likelihood and impact (a.k.a. severity).

The fragment of a DDP screen shown in Fig. 4 conveys some of the richness of its interface.

3. Application of DDP to study technology infusion of an advanced V&V technique

This section considers the problem of infusing an advanced V&V technique into mainstream use.

There is a continuing need to advance the state of the practice of V&V by making use of new and emerging V&V techniques. One source of promising such techniques is the computer science research community. However, technology infusion – turning emerging techniques into mainstream practices – is in general a challenging problem for many disciplines.

The DDP process described in the previous section has been used to good effect to plan for the infusion of novel technologies into use on spacecraft. The purpose of this section is to show how this same process can be used to explore the infusion of a V&V technique into use during software development.

This section uses the V&V technique of *model checking* as an illustration of this point.

3.1. Model checking for V&V

Model checking takes as input a system description (usually in the form of a finite state machine or machines) and a logical property about behaviors of that system (usually in the form of a temporal logic formula expressed over sequences of states in the system's behavior). Model checking does the equivalent of an exhaustive search of the state space of the system to ascertain whether or not the logical property is true of the system. For certain kinds of properties, when they are found to not hold, model checking can also return a counter-example illustrating property violation, in the form of a sequence of steps from initial state of the system to the point where the violation is reached. Model checking can be used to check both "safety" and "liveness" properties. The key to model checking is its use of computer science techniques, enabling it to scale to analysis of much larger systems than would be feasible with naïve exhaustive search of a large state space.

Several groups within NASA have conducted pilot studies of model checking, generally with very positive results. For example: Schneider used model checking to analyze the "Dual Redundant System" (part of the Cassini spacecraft's fault protection system), [Schneider at al, 1998]. He states, "Six separate requirements … were validated. Each of the six … involved exhaustive examination of approximately 100,000 states…". The NASA Ames Research Center's Automated Software Engineering Group http://ase.arc.nasa.gov/ used the SPIN model checker [Holzmann 1997] to find bugs in the intelligent plan execution software prior to its deployment in the Deep Space 1 spacecraft [Havelund et al, 2001].

Generally, the results of these and similar pilot studies have been promising, and research into this approach is expected to continue, both within NASA and in the broader software engineering community. However, there has been little transfer of this approach into standard practice for spacecraft flight code development and assurance. *Why is this?* The subsections that follow show the application of DDP to study this question.

3.2. DDP applied to the challenge of model checking infusion

The question of infusion of model checking technology is addressed in DDP by using its core concepts as follows:

- **Objectives** are used to represent the intended use of the model checking, including both purpose (what it is applied to, what aspects of V&V it is being used for) and performance (who will do it). For example, the Objective might be to have test engineers apply model-checking technology to during integration testing.
- **Risks** are used to represent factors that impede attainment of the infusion Objectives. For example, whoever is applying model-checking might lack the expertise to specify the properties to be checked for in the model checker's notation (typically, some form of temporal logic).
- **Mitigations** are used to represent possible approaches to overcoming those Risks (e.g., develop and give training in the use of temporal logic for property specification).

To do the study, information required to populate DDP was gathered in sessions involving model-checking experts who have experienced first-hand the excitement and the challenges of applying model checking to real-world problems.

The subsections that follow provide some further detail of the kind of information gathered in the course of this study, and the use of that model to begin investigation of its ramifications. This study is not yet complete, but the information is illustrative of the approach.

3.3. Populating DDP with details of model-checking infusion

To do this study, the information to populate DDP was gathered in sessions involving model-checking experts who have experienced first-hand the excitement and the challenges of applying model checking to real-world problems.

3.3.1 Objectives

Two groups of Objectives were considered – the first group encompassing the choices of artifacts to which model checking could be applied, and the second encompassing the choice of people who would apply model checking. For example, to explore the case of model checking of requirements (e.g., to validate a system's requirements against safety properties), done by the developer of the system itself, only the two Objectives "1.1.3:validation" and "2.1:developers" would be of concern. DDP allows individual Objectives to be turned on or off from consideration.

This Objective tree is shown below.

1:artifacts
- 1.1:rqmts
  - 1.1.1:consistency
  - 1.1.2:completeness
  - 1.1.3:validation
  - 1.1.4:test case generation
- 1.2:design
  - 1.2.1:requirements verification
  - 1.2.2:bug finding
- 1.3:code
  - 1.3.1:requirements verification
  - 1.3.2:unit testing
  - 1.3.3:integration testing
  - 1.3.4:structural, defect detection
  - 1.3.5:functional errors, bug finding
  - 1.3.6:timing errors
- 1.4:models
  - 1.4.1:sanity checking
  - 1.4.2:validation

2:who uses the tool
- 2.1:developers
- 2.2:test engineers
- 2.3:QA
- 2.4:IV&V
- 2.5:model checking gurus

3.3.2 Risks

Possible issues that might impede the use of model checking were captured as DDP's "Risks". Two main categories were considered – *technical* issues that impede use of model checking (e.g., state space explosion), and the *social* issues that impede use of model checking (e.g., resistance to learning new languages and tools, as would be required in most ways of using model checking).

The categorization is used to spur thinking of the full range of problems, and group items to facilitate navigation and scrutiny. It is not intended to serve as a general-purpose taxonomy for use beyond this one study.

1:Technical issues
- 1.1:state space explosion
- 1.2:slow turnaround time
- 1.3:notation that mc can't handle
  - 1.3.1:design notation incompatible with model checking
  - 1.3.2:property notation incompatible with model checking
- 1.4:challenging generation of environment models
- 1.5:unknown what applications domains are suitable
- 1.6:unknown how much work it takes
- 1.7:complexity of deciphering error traces

2:Social Issues
- 2.1:resitance to learning new languages and tools
- 2.2:need to have specification expertise
  - 2.2.1:modeling expertise (how to build the model)
  - 2.2.2:LTL etc expertise (how to specify the properties)
- 2.3:well documented requirements are lacking
- 2.4:beneficiaries not the ones who do it
- 2.5:large effort of applying model checking
- 2.6:knowledge of the application domain is required

3.3.3 Mitigations

Mitigations are the possible activities that could reduce the adverse impact of Risks, and thereby lead to increased use of model checking. The ones considered are listed below. These have not been arranged into any major categories, other than the small subtree beneath "increase computing resources".

1:tools for abstraction
2:tools for translation into mc Ls
3:hiring PhDs
4:training application engineers
5:increase computing resources
- 5.1:chips get faster
- 5.2:more memory
- 5.3:parallel h/w

6:short training course for LTL el al
7:emphasize the unique role that m/c can play
8:cost of failure a driver
9:specification patterns for properties
10:model checking provided as a "service"
11:Develop cost models
12:Case studies
13:Baselining & benchmarking
14:(Funded) partnerships with projects
15:Search heuristics
16:custom model checkers for programming Ls
17:compositional m/c
18:marketing
19:design for verification
20:tools for visualizing results
21:include mc into existing toolset
22:pick customers

3.3.4 Quantitative relationships ("effects" and "impacts")

Gathering the quantitative relationships between the Objectives and Risks, and between the Risks and Mitigations, is generally the most interesting and time consuming part of information elicitation when following the DDP process.

A portion of this kind of information for the model-checking infusion study is shown in Figure 5, a partial screenshot taken from the DDP tool. The quantitative relationships are captured in a matrix whose rows correspond to Mitigations, and columns to Risks. A cell entry indicates the strength of the effect of the row Mitigation at reducing the column Risk. This is usually expressed as a number in the range [0, 1] where the extreme of 0 means no effect whatsoever, and the extreme of 1 means completely effective (i.e., eliminates the Risk). A blank cell is equivalent to an entry of 0. An intermediate value, k say, means it reduces the Risk by the proportion k. For example, the value 0.99 in cell at the intersection of the Mitigation row "training application engineers", and the Risk column "property notation incompatible with model checking" means the application of that Mitigation will reduce by 99% that Risk. The assumption underpinning this high value is that the aspects of LTL or similar formal property notations needed for model checking can be very effectively taught to practitioners.

On occasion, a Mitigation may actually make the situation *worse*. This is indicated by giving a negative number, in the range [-1 0), as the strength of the effect. The magnitude of this negative number indicates the likelihood of *inducing* the Risk. For example, there is a value of –0.3 in the topmost white row (Mitigation: "tools for abstraction" and Risk "complexity of deciphering error traces", on the grounds that abstraction moves a specification further from the system, rendering deciphering of error traces that result from model checking somewhat *more* problematic.

The kinds of numbers visible in the fragment of the "effect" matrix of Figure 5 are representative of those we see entered for assessments of advanced technologies. The nature of these assessments precludes high precision entries. The lack of long-term experience from which to extract statistical measures forces the need to make use of experts' estimates, and this low level of precision is what we must work with. Nevertheless, we find that the aggregation of these coarse estimates can guide decision-making.

The information that connects Objectives to Risks is captured in a similar manner in the DDP "impacts" matrix (in the interest of brevity, not shown here).

Populating these matrices is done in a group setting, with all the experts present. As this proceeds, they may think of additional items (e.g., another Risk), in which case these get added in on the fly and the process continues, with those new items now included. Disagreement among those experts about a numerical value does occur, and usually indicates they are thinking of different circumstances. It can therefore be resolved by refining to greater detail (e.g., decomposing a Risk into two), thus allowing each expert's estimate to be

		REs	[-]Technical issues								[-]Social Issues				
		REs	state space explos	slow turnaro time	[-]notation that mc can't handle		challer genera of enviror models	unknov what applic domair are	unknov how much work it takes	comple of deciph error	resitan to learnin new langua	[-]need to have specification expertise		well docum require are	b n c
		REs			design notatio incomp with model	proper notatio incomp with model						modeli experti (how to build	LTL etc experti (how to		
SOs	SOs	FoM\R	1	1	1	1	1	1	1	1	1	1	1	0.97	1
hiring PhDs		9.9	0.8	0.1	0.9	0.9	0.8	0.6	0.3	0.99	0.9	0.8	0.99	0.1	
training application engineers		11.2	0.3	0.7	0.9	0.99	0.9	0.9	0.3	0.8	0.1	0.99	0.99	0.5	0
[-]increase computing resources	chips get faster	1.14	0.1	0.7						0.1					
	more memory	1.2	0.6	0.3											
	parallel h/w	1.82	0.7	0.7											
short training course for LTL el al		2.03				0.99							0.99	0.2	0
emphasize the unique role that m/c		1.39						0.7			0.3				
cost of failure a driver		1.39						0.7			0.3				
specification patterns for properties		2.05		0.1		0.7					0.3		0.7	0.3	0

Figure 5. Quantitative relationships between Mitigations (rows) and Risks (columns)

incorporated, in its proper place.

3.4. Using the DDP Model

Once the DDP model has been populated, it can be used to explore the ramifications of the combined set of information, and ultimately to make decisions. A brief example scenario follows.

Suppose that the objective is to understand the use of model checking for validation of a system's requirements, to be done by the developers of the system itself. In the DDP tree of Objectives, only the two "1.1.3:validation" and "2.1:developers" would be turned on for consideration.

As a starting point, suppose none of the Mitigations were chosen – what would the risks be? The DDP tool automatically computes the risk measures, and offers several ways to scrutinize the results. A key visualization is shown in Figure 6. Each bar corresponds to a Risk, and

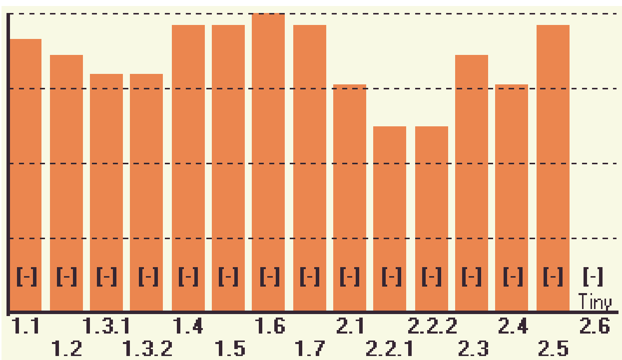

Figure 6. Risk magnitudes

its height corresponds to that Risk's contribution to risk. In this and other DP bar charts, the vertical axis is a *log* scale. In the figure, the faint dashed horizontal lines demark levels that are approximately a factor of 2 apart, so the difference between the tallest bar in this figure (labeled 1.6 at its base) and the bar two to its right (labeled 2.1 at its base) is approximately a factor of 2. This tallest magnitude Risk bar is 1.6:unknown how much work it takes. The smallest magnitude Risk – the rightmost column, labeled 2.6 – is so small that it doesn't even show up as a bar! This is the Risk "knowledge of the application domain is required". It is so low because the system developer, who already has that knowledge, is doing the application, so this Risk has very little impact on either of the Objectives selected.

One of the Mitigations that is very effective against the tallest risk is "14:(Funded) partnerships with projects". If this Mitigation is selected, the Risk 2.6 is considerably reduced. In fact, that one Mitigation considerably reduces quite a number of Risks. This is evident in the DDP recalculation and redisplay following that one selection, seen in Figure 7. The green segments denote portions of Risks that have been reduced by the selected Mitigations. The widespread appearance of green following that one selection shows its beneficial effects!

Suppose, however, that this Mitigation is infeasible for some reason (e.g., insufficient money is available to fund

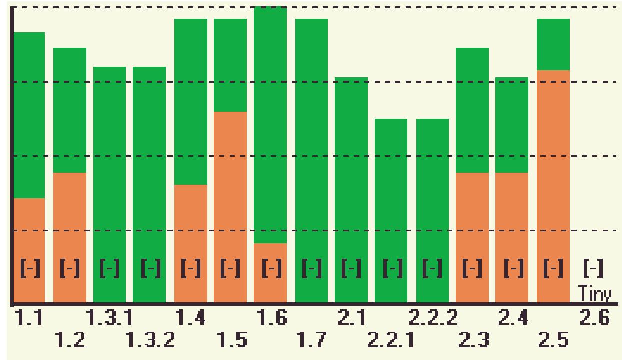

Figure 7. Risk magnitudes, when a key Mitigation adopted.

a project's collaboration in this manner). We would need to find alternate ways to reduce risk. One of them is to select the Mitigation "11:Develop cost models", whose risk reducing effects are shown in Fig. 8.

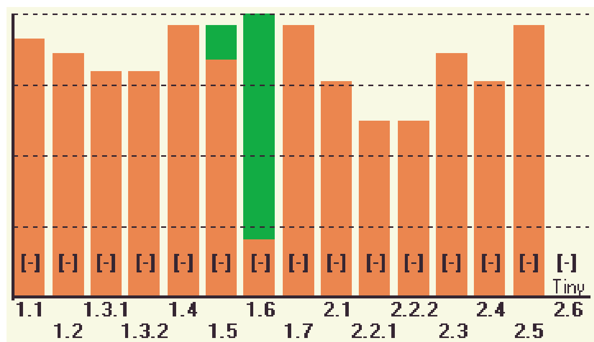

Figure 8. Risk magnitudes, when a lesser Mitigation adopted.

While this effectively reduces Risk number 1.6, it has little effect elsewhere (only a small effect on Risk number 1.5). Thus there is a need to select some further Mitigations that address the other high magnitude Risks.

The situation after selection of three Mitigations:

- "11:Develop cost models",
- "4:training application engineers" and
- "2:tools for translation into mc Ls" (the experts meant by this cryptic title "tools for translation into model checking languages", e.g., [Bose, 1999], [Mikk et al, 1998]

is shown in Figure 9.

The two large magnitude Risks remaining at this point are

"1.1:state space explosion" and
"2.3:well documented requirements are lacking".

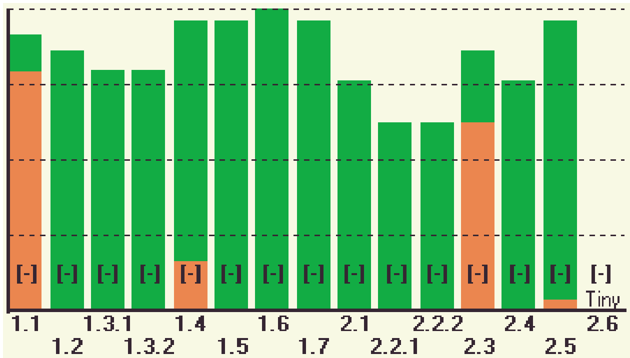

Figure 9. Risk magnitudes, with three Mitigations adopted.

State space explosion is often cited as a limiting factor for model checking, so it is not surprising to see it as a major Risk (i.e., impediment to infusion) at this point, especially since none of the selected Mitigations did much to address it. However, it was but one of several high-magnitude impediments. The Mitigations that reduced those other impediments would not, generally, have been solved by a narrow focus on the state space explosion issue.

The need for well-documented requirements is something that is not so often cited as a problem, but does arise in practice. Indeed, this was more problematic than initially expected in a model checking study in which I was involved [Feather et al, 2001a].

3.5. Observations on use of DDP for studying infusion

The previous subsections have illustrated the DDP approach to studying the infusion problem. The illustrative example was of infusing model checking as a V&V technology. The modest amounts of data gathered so far in the course of this study suffice to elucidate some aspects of the infusion challenge. Elaboration of this information is expected to yield more detailed insights, especially by focusing more on the Objectives of a specific area of application.

Applications of DDP to study the infusion of (non-V&V) technologies have generally had such a mission-specific bent. Typical numbers of concepts for these studies are Objectives, 30 – 50; Risks, 30 – 70; Mitigations, 50 – 100. To gather this amount of information generally takes four half-day meetings each with 10-20 experts involved. Thus the effort of running a DDP study is not trivial – the primary expense is the time of the experts who provide the information and to make the decisions. A facilitator is needed – someone who both understands the DDP process, and has a feel for the broad range of concerns that the study must deal with. The facilitator guides the elicitation and decision making steps. The DDP tool is run throughout the sessions, its screen projected and visible to all the participants. As information is gathered, it is entered into the tool in real time. Switching among the various ways of presenting information supports decision-making. Someone conversant with the DDP tool controls the tool, does data entry, etc. In some studies, the same individual has acted as both facilitator and tool controller; in others, separate individuals have filled these two roles.

Anecdotal evidence culled from these applications is supportive of the value of DDP. Initially skeptical participants typically emerge convinced that DDP has helped. [Cornford et al., 2001] reports benefits of:

- Clarification of a customer requirement leading to considerable savings in work not required.
- Rejuvenation of a technology by identification of opportunities for its utilization.
- Support for adoption of a commercial software development environment (balancing the pros and cons of making this switch from current practice)

For these technology infusion studies, the main benefits accrue from pinpointing the most critical areas within a large space of concerns, and guiding experts toward superior alternatives. The overall perception is that the benefits of improved decision-making in these early stages are well worth the effort invested.

4. Application of DDP to selecting V&V activities

The second theme of this paper is on the *selection* of V&V activities. For most projects, there are many more V&V activities that could be applied than there are resources to pay for. Hence the selection of just which to apply, and to which parts of the software to apply them, can be very challenging. This section summarizes the ways in which DDP has been applied and extended to approach this problem.

4.1. Representing software V&V concerns in DDP

The key insight for representation of software V&V concerns is seen in Cornford's original vision (Fig. 1, earlier): *the purpose of V&V activities is to reduce risk.*

Two things should guide selection of those activities: their benefits (reduction of risks), and their costs (resources it takes to apply them). The DDP model of Objectives, Risks and Mitigations matches the V&V selection problem as follows:

- Requirements of the software being V&V'd are captured as DDP's "Objectives". These can and should include both *product* and *process* requirements. Product requirements encompass what the software should accomplish and what constraints there are on its operation (e.g., disk, memory, CPU). Process requirements encompass how it is to be developed and what constraints there are on its development (e.g., time, budget).

- Risks that V&V might be called upon to help reduce are captured as DDP's "Risks". Again, these can and should encompass both product and process risks. Examples of product risks are that the software computes an erroneous result, or runs too slowly. Examples of process risks are that the requirements are misunderstood, or the design phase takes too long.
- The gamut of possible V&V activities is captured as DDP's "Mitigations". Once again, these should be wide ranging, encompassing activities that might be employed to reduce any of the kinds of the risks. For example, core V&V activities such as reviews, inspections, analyses and tests can be applied to directly improve the quality of the software artifacts themselves. Practices such as establishing coding guidelines, requiring formal approval of the management plan, maintaining a change control board to handle change requests, can be applied to improve the quality of the development process.

Various experiments to represent software V&V activities in the DDP framework are summarized in the subsections that follow.

4.2. Relating software risks to software development best practices

A broad-ranging taxonomy of software development risks is found in the Software Engineering Institute's Software Risk Evaluation Method [Sisti & Sujoe, 1994]. These are relevant to a wide range of software development efforts. Our group entered these as DDP "Risks". A fragment is shown in Figure 10.

In a similar manner, we entered activities from Level 2 ("Repeatable") of the SEI Capability Maturity Model for Software, CMM v1.1 [Paulk et al, 1993] as DDP "Mitigations".

In the absence of reported data on the effectiveness of those activities at mitigating those risks, our group made estimates of these. In a preliminary study, we began by simply estimating which risks were mitigated to some extent by which activities [Feather et al, 1999]. We further refined this information by making quantitative estimates of how much each activity reduced each risk [Cornford et al, 2000].

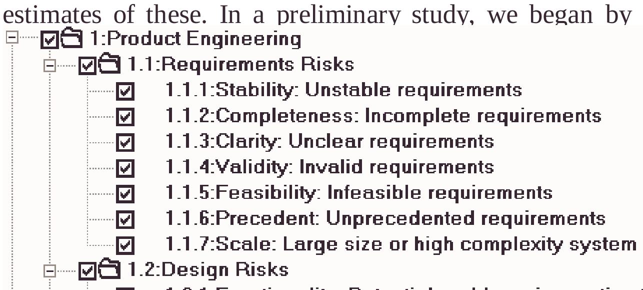

Figure 10. CMM Level 2 Practices as DDP Risks

This overall approach was then elaborated to give it a NASA-specific focus, by substituting a NASA taxonomy of development practices in place of the CMM activities. Again, these were entered these into DDP as "Mitigations", and estimates made of their effectiveness. The taxonomy we adopted was that used by another NASA tool, "Ask Pete". That tool is used to (among other things) help make recommendations of which Software Quality Assurance practices to perform, and provide detailed guidance, project planning, etc. Further details on the Ask Pete, including a download of the tool itself, see:

http//osat-ext.grc.nasa.gov/rmo/pete/index.html

We arranged to have Ask Pete and DDP exchange information, so that they could be used together in the following 3-step process:

1. Ask Pete is run to query the user of the characteristics of the software development effort. Based on the user's answers, the Ask Pete tool makes an initial recommendation of software quality assurance (SQA) practices to perform. This information is then transferred to DDP.
2. DDP is used to tailor this initial recommendation to the particular situation. DDP is pre-loaded with our estimates of the effectiveness of the SQA practices at reducing software development risks. The user can tailor this information by ranking the risks, adding new ones if need be, and by adjusting the effectiveness estimates to match their skill set on hand. On the basis of this tailored information, they may then adjust the selection of SQA practices. This information is then transferred back to Ask Pete.
3. Ask Pete is used to generate final reports and documentation, taking into account the recommendations as changed by DDP.

Either tool can be used in isolation. In conjunction, they support each other in the manner outlined above. Further details of this collaboration are reported in [Kurtz & Feather, 2000].

4.3. Risk and cost elaborations of the DDP model

There are important nuances of risks and costs (especially *software* risks and costs) that have required elaboration to the DDP model. Briefly, these are:

Representing the time at which Mitigations are performed: Large software development efforts may span several years, and budgeting for them is non-trivial. Assigning Mitigations to the time phase in which they are to be performed allows for DDP to match expenditure over time to the availability of funding. A further benefit is that the information also provides key insight into the "risk profile" – how risk diminishes over the course of the planned development, as discussed in (Cornford et al., 2002). Plans that reduce risks early are, in general, preferred over plans that attain the same final risk level but do so by reducing risks late. The reason is that all of these plans contain considerable uncertainty (remember, DDP is applied early in the lifecycle where solid information is lacking). A plan that reduces risk early can slip and still have reduced risks to tolerable levels by the originally planned launch date (Plan A in Figure 11). The same tolerance to slippage is not true of a plan that reduces risk late (Plan B in Figure 11).

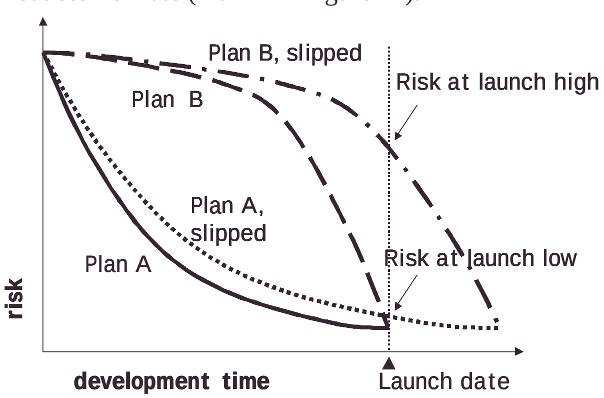

Figure 11. Risk profiles and slippage

Representing the cost of repairing a problem (e.g., fixing a bug): Costs are associated with Mitigations. However for the "detection" kind of Mitigations there is both a cost of performing it (e.g., the cost of running a unit test) and a cost of performing the repair (e.g., fixing a bug uncovered during unit test). The DDP model has been elaborated to associate a "repair" cost with a Risk.

Repair cost escalation: There is an interaction between when Mitigations are performed, and what it costs to repair detected Risks. The repair cost can depend on the time phase in which the detected Risk is repaired. Matching time phases to development phases (e.g., requirements, design, unit test, system test, deployment), allows representation of the well-known cost escalation of fixing a flaw later in the lifecycle (for a discussion on this and related issues, see [Shull et al, 2002].

Representing the potential for Mitigations to cause problems as well as reduce them: Mitigations can be asserted to have an effect of *increasing* the likelihood of Risks. Again, there is an interaction with the elaboration of when Mitigations are performed – clearly, the problems introduced by a Mitigation can be detected and repaired only by Mitigations performed later.

All the above elaborations have been incorporated into the DDP model. The net result is the capability to represent many of the nuances of cost and risk.

As a small case study to illustrate this point, DDP was used to represent the scenarios of different mixes of inspection and testing presented by Gary Gack, in the "Defect Tracking + Inspections = $ in Your Pocket" portion of his website

http://www.iteffectiveness.com/defecttracking.html.

Gack's calculations show the value of aggressive use of inspections and structured testing as capable of leading to a better final product (fewer defects) at less cost. This occurs because earlier-phase activities (e.g., inspections) lead to the early discovery, and therefore inexpensive correction, of problems that would otherwise be uncovered only in later phases (e.g., testing) when repair costs are higher. Cost considerations such as these are also discussed in [Kaner, 1996]. The DDP representation of Gack's model encompasses all the details of his calculations [Feather et al, 2001b].

To date, most DDP applications have been at the system-wide level, rather than on software-specific planning. One major DDP study that did have a software-intensive focus addressed adoption of a ground-based software development environment for in-flight use. That study's Risks encompassed both software product and software process risks. Prior to the study, there was general agreement on the potential benefits, but not a clear understanding of the risks. The net result was the confidence to go ahead with the effort, based on an understanding of the relative pros and cons, along with a detailed plan for how to address the identified risks.

5. Conclusions and related work

This paper has summarized DDP, a process for risk assessment and risk mitigation planning. DDP has been in use at JPL for guiding technology selection and technology infusion. The focus of this paper is on DDP's potential for application to specifically software V&V concerns. Model checking applied to V&V was used as an illustrative example of an infusion challenge. Software development practices, and their representation in DDP, were discussed in the context of V&V selection.

The hallmark of DDP is its focus on early-lifecycle decision-making. Its foundation is a quantitative model that links "Objectives" to the "Risks" that threaten them, and links "Mitigations" to the "Risks" that they reduce. The DDP process calls for the elicitation of this information from expert users, followed by their decision making guided by the combination of this information.

Custom DDP software supports all phases of this process.

5.1. What alternatives are there to a DDP-like approach?

One extreme is to rely on past experience and/or human insight to make the decisions, not bothering with decision support processes or tools. In cases where the problem at hand is similar to past problems, this may suffice. A highly skilled individual who comprehends all the relevant concerns, and is able to fathom the relative costs and benefits of various alternatives, may not need any help. However, in novel and challenging situations, past experience and human insight do not necessarily lead to anywhere near optimal decisions.

Qualitative approaches, supported to varying degrees by a variety of methods, processes and tools, work well as an adjunct to human decision-making. They encourage the elicitation of issues, and excel in providing semi-formal means to record interrelationships among those issues. Examples of qualitative approaches include:

- Quality Function Deployment (QFD) [Akao, 1990] for design in general.
- The i* approach [Chung et al, 1999], with tool support [Tran & Chung, 1999] has been used to exploring a modest number of major design alternatives in [Mylopoulos et al, 2001].
- The WinWin process [Boehm et al, 1994] and tool-supported applications [In et al, 2001] to study software requirements. Somewhat similar in spirit is the cost-value approach to requirements prioritization in [Karlsson & Ryan, 1997].
- The KAOS method with its focus on goal decomposition to lead from overall "goals" to system-specific requirements [Letier & van Lamsweerde, 2002], for which there is also tool support [Bertrand et al, 1998].

The idea of a focus on the mapping between requirements (in DDP terminology, "Objectives"), problems (in DDP terminology, "Risks") and solutions (in DDP terminology, "Mitigations") recurs in many of the qualitative approaches. As well as the work cited above, [Raz & Shaw, 2000] study "fault mappings"; [Alexander, 2002] uses "threatens", "mitigates" etc. relationships among elements of so-called "misuse cases"; [Kozaczynski, 2002] links Requirements to Risks, and Risks to "Architecture" (things that "mitigate" Risks).

These qualitative approaches work well to promote brainstorming of ideas, and to help users select from among a relatively small number of alternatives. DDP too facilitates brainstorming, and, because of its *quantitative* basis, seems to also work well in cases where there are relatively large numbers of alternatives. Experts have found DDP useful to guide their selection of a suite of tasks out of a pool of many dozens, even hundreds, of them.

DDP's quantitative basis also enables use of heuristic search techniques to locate near-optimal solutions [Feather & Menzies, 2002]. These techniques are used to guide users towards interesting solution areas, and are not intended to fully automate decision-making. Our experience to date suggests the value of iterations between DDP tool searches and experts. On the basis of the experts' initial information, the tool identifies promising solutions; the experts then critique these solutions, often revealing additional information (e.g., "we can't do those two Mitigations together because …"). This information is incorporated, and the search re-run, etc.

Design-centric approaches to risk *assessment* are relatively mature. For example, the nuclear power industry (e.g., [INSC, 1998]) used probabilistic risk assessment. In particular, Fault trees [Vesely et al, 1981] are commonly used in analysis of hardware designs, and there are many efforts to adapt these approaches to software systems (e.g., Software Fault Tree Analysis [Leveson, 1995], and Software Failure Modes and Effects Analysis). Generally these approaches presuppose the existence of a detailed design, which precludes their use in the earliest stages of planning. There are efforts to extend their use to earlier phases, e.g., an application to requirements is described in [Lutz & Woodhouse, 1997].

There are, of course, many formal methods founded upon deep theoretical underpinnings and embodied in tools. These are used for analysis of specific properties of a given software design or implementation.

Generally, the purpose of these approaches is to validate/verify an existing design. In contrast, DDP's purpose is focused more on planning in advance the overall validation or verification effort.

Process-centric approaches are used to model the entire software development process, including use of V&V techniques. For example, [Stutzke & Smidts, 2001] use a stochastic model that takes into account where errors are made, and how they are detected and repaired. Another example is COQUALMO [Chulani 1999] (see also http://sunset.usc.edu/research/coqualmo/), which is "an estimation model that can be used for predicting number of residual defects … in a software product".

An overview of this kind of work is to be found in [Shull et al, 2002]. The parametric and stochastic reliability models used in these approaches are typically more sophisticated than that of DDP. However, they do not generally attempt to make the kinds of problem-specific distinctions that DDP allows for via its user-populated sets of Objectives, Risks, and Mitigations.

5.2. Challenges

Lack of empirical data is a recurring challenge when seeking to apply almost any quantitative approach to V&V planning. Such (unknown) data includes rates at which defects are introduced during development phases, efficacy of and effort to prevent or detect defects, costs of repairing defects, and impacts of defects.

The Center for Empirically-Based Software Engineering (CeBASE – http://www.CeBASE.org) was established to gather such knowledge. In the absence of such data, experts' estimates are a necessary substitute. DDP, intended for use in novel and challenging situations that do not fit the norm, must inevitably continue to rely heavily on such estimates.

Some approaches treat uncertainty explicitly. They attach confidence measures, and/or allow users to indicate ranges of values, and possibly distributions, for estimates. Sensitivity analysis is often used to reveal the variance of the computed results in terms of variance of the inputs. In DDP, a simple form of this is used to rank the criticality of the experts' quantitative estimates. Those estimates that, if changed, lead to the greatest variance in computed Objective attainment, are ranked highest criticality.

The overall objective is to make better decisions. Uncertainty is not necessarily an impediment to decision making. This line of work is explored in [Menzies et al, 2002], a small part of which is based on experiments with DDP models.

All the approaches cited have their strengths. It would desirable to blend the best that each of the different approaches to risk assessment and risk mitigation planning has to offer.

It could be that the greatest challenge of all, however, is not the need to infuse new V&V practices, or better plan V&V, but simply to increase adoption of already-known best practices. Perhaps the equivalent of the FDA's "Food Pyramid" (http://www.nalusda.gov/fnic/Fpyr/pyramid.gif) is needed for V&V - see Figure 12.

6. Acknowledgements

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

Burton Sigal (JPL), John Kelly (JPL), Jim Kiper (Univ. of Miami in Ohio), Tim Kurtz (NASA Glenn), Ken McGill (NASA IV&V), Tim Menzies (West Virginia University), Charles Norton (JPL), Peter In (Texas A&M) and Martha Wetherholt (NASA HQ), have had a positive influence on this work. At JPL Steve Cornford leads the entire DDP effort, to which Julia Dunphy, Mark Gibbel, Tim Larson, Kelly Moran and Jose Salcedo have made major contributions.

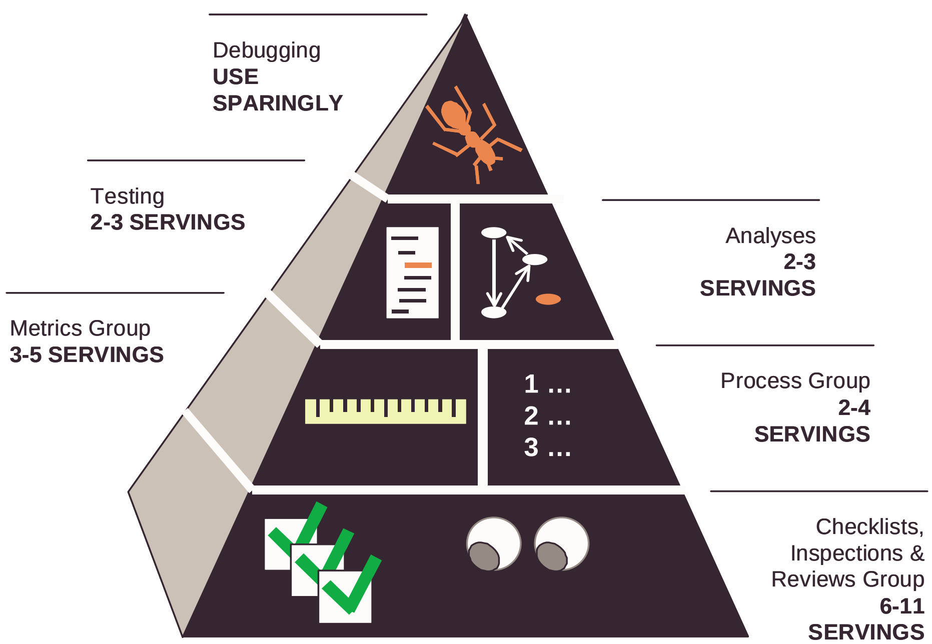

Figure 12. Hypothetical V&V Pyramid

8. References

[Akao, 1990] Y. Akao: "*Quality Function Deployment*", Productivity Press, Cambridge, Massachusetts, 1990.

[Alexander, 2002] I. Alexander: "Initial Industrial Experience of Misuse Cases in Trade-Off Analysis", *Proceedings of the IEEE Joint International Requirements Engineering Conference*, Essen, Germany, Sept. 2002, pp. 61-68.

[Bertrand et al, 1998] Bertrand, P., Darimont, R., Delor, E., Massonet, P., and van Lamsweerde, A.: "GRAIL/KAOS: an environment for goal driven requirements engineering", *Proceedings, 20th Int. Conference on Software Engineering*, 1998

[Bose, 1999] P. Bose: Automated Translation of UML Models of Architectures for Verification and Simulation Using SPIN. *Proceedings of the 14th IEEE International Conference on Automated Software Engineering*, pp. 102-109, Cocoa Beach, Florida, October 1999.

[Boehm et al, 1994] B. Boehm, P. Bose, E. Horowitz and M. Lee.: "Software Requirements as Negotiated Win Conditions", *Proceedings 1st International Conference on Requirements Engineering*, Colorado Springs, Colorado, pp. 74-83, 1994.

[Chulani 1999] S. Chulani: "COQUALMO (COnstructive QUALity Model) A Software Defect Density Prediction Model", in *Project Control for Software Quality*, editors: R. Kusters, A. Cowderoy, F. Heemstra & E. van Veenendaal, Sharker Publishing 1999.

[Chung et al, 1999] L. Chung, B.A. Nixon, E. Yu and J. Mylopoulos: *Non-Functional Requirements in Software Engineering*, Kluwer Academic Publishers, Boston, Oct 1999.

[Cornford, 1998] S.L. Cornford: "Managing Risk as a Resource using the Defect Detection and Prevention process", *Proceedings, 4th International Conference on Probabilistic Safety Assessment and Management*, 13-18 September 1998, New York City, NY, International Association for Probabilistic Safety Assessment and Management.

[Cornford et al, 2000] S.L. Cornford, M.S. Feather, J.C. Kelly, T.W. Larson, B. Sigal & J.D. Kiper: "Design and Development Assessment", *Proceedings, 10th IEEE International Workshop on Software Specification and Design*, San Diego, California, 5-7 Nov 2000, pp 105-204.

[Cornford et al, 2001] S.L. Cornford, M.S. Feather and K.A. Hicks: "DDP – A tool for life-cycle risk management", *Proceedings, IEEE Aerospace Conference*, Big Sky, Montana, Mar 2001, pp. 441-451.

[Cornford et al, 2002] S.L. Cornford, J. Dunphy, and M.S. Feather: "Optimizing the Design of end-to-end Spacecraft Systems using risk as a currency", *IEEE Aerospace Conference*, Big Sky, Montana, 2002

[Feather et al, 1999] M.S. Feather, J.C. Kelly and J.D. Kiper: "Prototype Tool Support for SEI Process and Risk Knowledge", *NASA 2nd Annual Workshop on Risk Management*, Morgantown, West Virginia, October 1999.

[Feather et al, 2000a] M.S. Feather, S.L. Cornford and M. Gibbel: "Scalable Mechanisms for Requirements Interaction Management", *Proceedings 4th IEEE International Conference on Requirements Engineering,* Schaumburg, Illinois, 19-23 Jun 2000, IEEE Computer Society, pp 119-129

[Feather et al, 2000b] M.S. Feather, S.L. Cornford and T. Larson: "Combining the Best Attributes of Qualitative and Quantitative Risk Management Tool Support", *Proceedings, 15th IEEE International Conference on Automated Software Engineering*, Grenoble, France, 11-15 September 2000. IEEE Computer Society, pp 309-312.

[Feather et al, 2001a] M.S. Feather, B. Sigal, S.L. Cornford and P. Hutchinson: "Incorporating Cost-Benefit Analyses into Software Assurance Planning", *Proceedings, 26th IEEE/NASA Software Engineering Workshop*, Greenbelt, Maryland November 27-29.

[Feather et al, 2001b] M.S. Feather, S. Fickas and N.-A. Razermera-Mamy: "Model-checking for validation of a fault protection system", Proceedings, 6th IEEE International Symposium on High Assurance Systems Engineering, 2001, pp. 32-41.

[Feather & Menzies, 2002] M.S. Feather and T. Menzies: "Converging on the Optimal Attainment of Requirements", Proceedings *of the IEEE Joint International Conference on Requirements Engineering*, Essen, Germany, Sept. 2002, pp. 263-270.

[Greenfield, 1998] M.A. Greenfield: "Risk Management: 'Risk As A Resource' " http://www.hq.nasa.gov/office/codeq/risk/

[Havelund et al, 2001] K. Havelund, M. Lowry and J. Penix: "Formal analysis of a space-craft controller using SPIN", *IEEE Transactions on Software Engineering*, 27(8), Aug 2001, pp. 749-765.

[Holzmann, 1997] Holzmann, G.J., "The Model Checker SPIN": *IEEE Trans. on Software Engineering*, Vol. 23, No. 5, May 1997, pp. 279-295.

[In et al, 2001] H. In, B. Boehm, T. Rodgers and M. Deutsch: "Applying WinWin to Quality Requirements: A Case Study", *Proceedings 23rd International Conference on Software Engineering*, Toronto, Ont., Canada, May 2001, pp 555-564.

[INSC, 1998] International Nuclear Societies Council: "Role of Risk Methods in The Regulation of Nuclear Power" in Action Plan 1997-1998 http://www2s.biglobe.ne.jp/~INSC/INSCAP/Risk.html

[Kaner, 1996]. C. Kaner: "Quality Cost Analysis: Benefits and Risks", *Software QA* Vol 3, #1, p. 23, 1996.

[Karlsson & Ryan, 1997] J. Karlsson and K. Ryan: "A Cost-Value Approach for Prioritizing Requirements", *IEEE Software*, Sept./Oct. 1997, pp 67-74.

[Kozaczynski, 2002] W. Kozaczynski: "Requirements, Architectures and Risks", ", *Proceedings of the IEEE Joint International Requirements Engineering Conference*, Essen, Germany, Sept. 2002, pp. 6-7.

[Kurtz & Feather, 2000] T. Kurtz and M.S. Feather: "Putting it All Together: Software Planning, Estimating and Assessment for a Successful Project", in *Proceedings of 4th International Software & Internet Quality Week Conference*, Brussels, Belgium, 2000.

[Letier & van Lamsweerde, 2002] E. Letier and A. van Lamsweerde: "Agent-Based Tactics for Goal-Oriented Requirements Elaboration", *Proceedings 24th International Conference on Software Engineering*, Orlando, May 2002, pp.

83-93.

[Leveson, 1995] N.G. Leveson: "*Safeware: System Safety and Computers*". Addison-Wesley, Reading, MA, USA, 1995.

[Lutz & Woodhouse, 1997] R. Lutz and R.M. Woodhouse: "*Requirements analysis using forward and backward search*". Annals of Software Engineering (3), 1997, pp. 459-475.

[Menzies et al, 2002] T. Menzies, E. Chiang, M.S. Feather, Y. Hu and J.D. Kiper: "Condensing Uncertainty via Incremental Treatment Learning", in submission to Annals of Software Engineering, special issue on Computational Intelligence; available at http://www.csee.wvu.edu/~menzies/pdf/02itar2.pdf

[Mikk et al, 1998] E. Mikk, Y. Lakhneck and M. Siegel: Implementing Statecharts in PROMELA/SPIN. *Proceedings of the 2nd IEEE Workshop on Industrial-Strength Formal Specification Techniques*, 90-101, Boca Raton, Florida, October 1998.

[Mylopoulos et al, 2001] J. Mylopoulos, L. Chung, S. Liao, H. Wang and E. Yu: "Exploring Alternatives during Requirements Analysis", *IEEE Software* 18(1), pp. 92-96, Jan-Feb 2001.

[Paulk, et al, 1993] Mark C. Paulk, Charlie Weber, Suzanne Garcia, Mary Beth Chrissis and Marilyn Bush: Key Practices of the Capability Maturity Model Version 1.1. *Technical Report CMU/SEI-93-TR-025*, Software Engineering Institute, Carnegie Mellon University, February 1993.

[Raz & Shaw, 2000] O. Raz and M. Shaw: "An Approach to Preserving Sufficient Correctness in Open Resource Coalitions", *Proceedings of the 10th International Workshop on Software Specification and Design*, San Diego, California, Nov. 2000, pp. 159-170.

[Schneider et al, 1998] F. Schneider, S.M. Easterbrook, J.R. Callahan and G.J. Holzmann: "Validating Requirements for Fault Tolerant Systems using Model Checking", *Proceedings, 3rd International Conference on Requirements Engineering*, April 1998.

[Shull et al, 2002] F. Shull, V. Basili, B. Boehm, A.W. Brown, P. Costa, M. Lindvall, D. Port, I. Rus, R. Tesoriero and M. Zelkowitz, "What We Have Learned About Fighting Defects", *Proceedings, 8th IEEE Symposium on Software Metrics*, 2002, pp. 249-258.

[Stutzke & Smidts, 2001] M. Stutzke and C.S. Smidts: "A Stochastic Model of Fault Introduction & Removal During Software Development", *IEEE Transactions on Reliability*, 50(2), June 2001, pp. 184-193.

[Tran & Chung, 1999] A. Tran and L. Chung: "NFR-Assistant: Tool Support for Achieving Quality", *Proceedings Application-Specific Systems and Software Engineering Technology*, Richardson, Texas, March 1999, pp 24-27.

[Vesely et al, 1981] W.E. Vesely, F.F. Goldberg, N.H. Roberts and D.F. Haasl: "*Fault Tree Handbook*", U.S. Nuclear Regulatory Commission NUREG-0492, 1981.